Getting Started with Git

Understanding Version Control Systems and Git

* Version Control Systems (VCS) maintain a history of changes to files for reference or rollback
* Collaboration tool
* Two components:
  + Centralized VCS – Relies on a master database to maintain all data, history, file revisions, etc…
* Git
  + Every person working on a project will have the full repository – Distributed VCS
  + Distributed VCS allows for a full clones of the entire historical record
  + Creating your first Repository
    - Repository – A collection of files that GIT tracks changes for
      * One directory with its subdirectories
      * The .git subdirectory makes it a repo
      * Can be created from scratch or cloned
    - #git config –global user.name “name”
    - #git config –global.email “email”
    - #git config –list
    - #git init
      * Initialize the current worker folder (NOTE: NOTHING IS TRACKED JUST YET UNTIL YOU STAGE IT)
    - #git add *folder name* or \*.\* to add everything inside
    - #git commit -m “*message here*” – to commit it and finally add the files you staged to the repository and be tracked by GIT
    - #git status – shows you changes or modifications
      * Green text means that changes are staged and ready to be committed
  + Files in Git have a 3-Staged Lifecycle
    - Unmodified
      * First added
    - Modified
      * When things are added or changed and it is now modified
    - Staged
      * When it is staged, it is now ready to be committed
      * Once a staged file has been committed, it revolves back to stage 1
  + Viewing Repository History
    - File timeline is called a ‘branch’
    - Initial is the original while the Head is where the next commit will be
    - Each commit will receive a Unique ID
    - #git log – will display the history of the repository
    - #git log –oneline – condensed version
    - #git log -n *number* – This will show n number of line/lines
  + Comparing Changes with Git Diff
    - #git diff – without any parameters will compare changes made between a single file
    - The minus means there has been something removed from the ‘a’ version and added to the ‘b’ version with the plus sign
    - The ‘@’ symbol is the chunk Identifier – The first two sets of numbers starting with a negative is from the ‘a’ version which displays the beginning line and the second number means the number of additional lines after, and the second two sets of numbers beginning with a plus displays the beginning line of the ‘b’ version
    - #git diff *two sets of the commit identifier –* Will compare the differences between two commits
  + Checking Out Previous Commits
    - Git checkout command reverts our working directory to match a specific commit
    - When making changes from the previous commits, the file will be detached from the master
    - #git checkout master – this will revert you back to the master branch
    - #git branch *new name* *unique identifier –* This will create a new branch and add the unique identifier to it
  + Git Reset and Revert to Undo Changes
    - #git commit –amend
      * Replaces the previous commit
      * Combines the old commit with a new one
    - #git reset
      * Removes the commit and reset the Head
      * #git reset head ~3
        + Will point the head 3 commits back
    - #git revert
      * Creates reverse commit, preserving all history
  + Untracking and Unstaging Files
    - #git restore –staged
      * Untrack or unstage a file
      * .gitignore
        + It’s a file that you can add files to, to ignore files
      * #git rm *filename* 
        + This will remove the whole file
        + Add - - *cached* operation remove from repository but will remain in the File System